続

実設計に応用できる演算回路スキルを身につけよう

デバイスの記事

外村元伸

第3回 拡大・縮小スケーリング・アルゴリズム演算の実装と評価

本連載第1回(本誌2006年7月号，pp.78-84)と第2回(本誌2006年8月号，pp.120-126)では，リニア補間法やキュービック補間法など，画像の拡大・縮小スケーリングに用いられるアルゴリズムを紹介した．今回は，それらのアルゴリズムを回路に実装するための注意点を解説する．実際にFPGAボードを使って，各種アルゴリズムとその演算器の評価を行った．　(編集部)

前回までで，画像の拡大・縮小スケーリング・アルゴリズムおよびそれらの演算器の構成方法についての解説がひととおり終わりました(1/5とその整数倍の演算器の構成例については，p.140のコラム「前回の質問の解答」を参照)．今回はそれらの演算器の実装についてお話しします．この実装の段階で，最後にもう一つの関門があります．

原画像を読み出してスケーリング表示するためには，まず拡大率または縮小率を指定し，表示画像と原画像のサイズの比を求めます．それから，一つの画素を表示するごとに座標の進みぐあいを把握して，次々と表示するための座標位置と補間に使用する参照画素の更新を制御する必要があります．**図1**にそのようすを示します．ここで，画素p_i

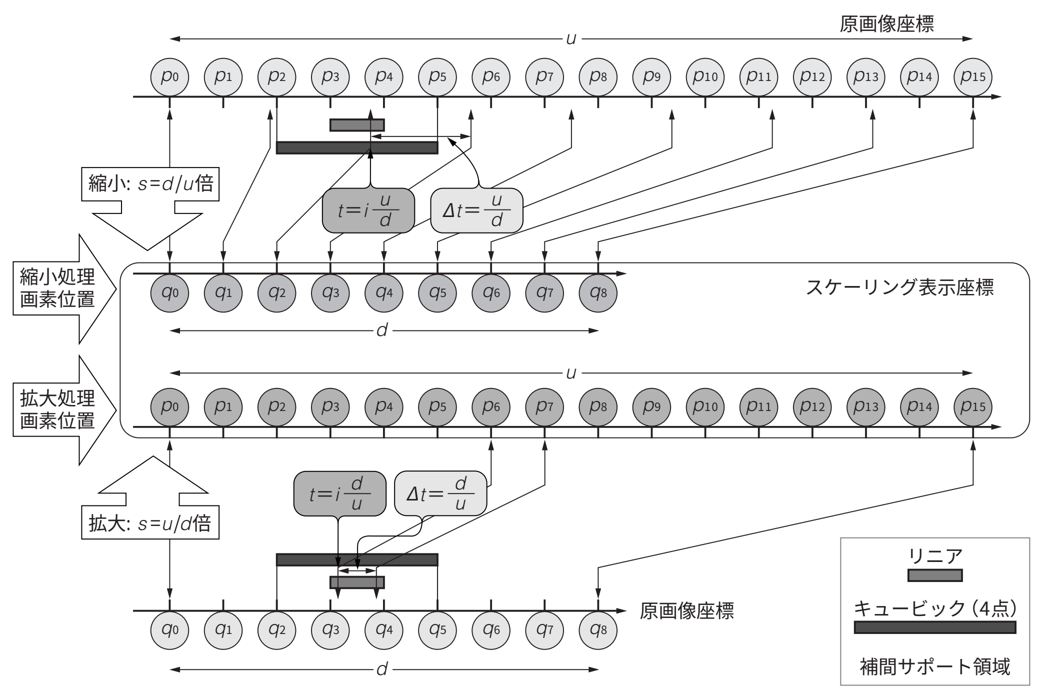


図1
縮小位置と拡大位置の画素値を求めるための補間候補画素点
図中央に示す画素列がスケーリング表示する位置座標系．図の上側が縮小の対象とする原画像の座標系で，図の下側が拡大の対象とする原画像の座標系．スケーリング表示の位置を原画像の座標位置に対応させて，補間を行う

KeyWord
リニア補間法，キュービック補間法，補間サポート，バイキュービック・スプライン，バイキュービック・コンボリューション，ゾーンプレート，三角関数演算，Bayer Pattern

の中心は座標位置 i にあるとします．**図1**の上側に縮小の場合を，**図1**の下側に拡大の場合を示しています．

以下にこの制御方法を説明します．



写真1　FPGAボードへの実装例

写真はデモンストレーション用にFPGAボード（英国Celoxica社製のRC300）でバイキュービック・スプライン（4×4画素参照）補間法を実現し，表示させたもの．タッチ・パネル式の液晶ディスプレイの裏側にFPGAボードがある．左上から原画，バイリニア，バイキュービックの順に拡大表示している

1．画素任意縮小/拡大スケーリングの制御

縮小する（**図1**の上側）にしても，拡大する（**図1**の下側）にしても，スケーリング表示する画素の座標位置が原画像のどこの位置に相当するのかを見つけ，その位置の画素の値を原画像からの参照画素補間によって求めます．

●1次元における補間の考えかた

そのためには，まずその位置を含む補間区間（$0 \leqq t < 1$）を把握します．そして補間区間は f_0 と f_1 の画素位置間にあるので，4点を参照する場合は，補間サポートと呼ぶ参照画素 f_{-1}，f_0，f_1，f_2 を相対的に取り出します．6点を参照する場合は，補間サポート f_{-2}，f_{-1}，f_0，f_1，f_2，f_3 を取り出します．ちなみに線形補間の場合は，補間サポートは f_0，f_1 のみです．

補間区間（$0 \leqq t < 1$）を制御するためには，スケーリング表示するときに，表示する隣接画素間のきざみ幅 Δt が，拡大率（u/d）または縮小率（d/u）によって決まるため，次の隣接画素に進むときに，Δt を加算し，$t \leftarrow t + \Delta t$ になりますが，もし $1 \leqq t + \Delta t$ になったら，$t \leftarrow t + \Delta t - 1$ として，補間サポートの参照画素位置を一つずつシフトします．

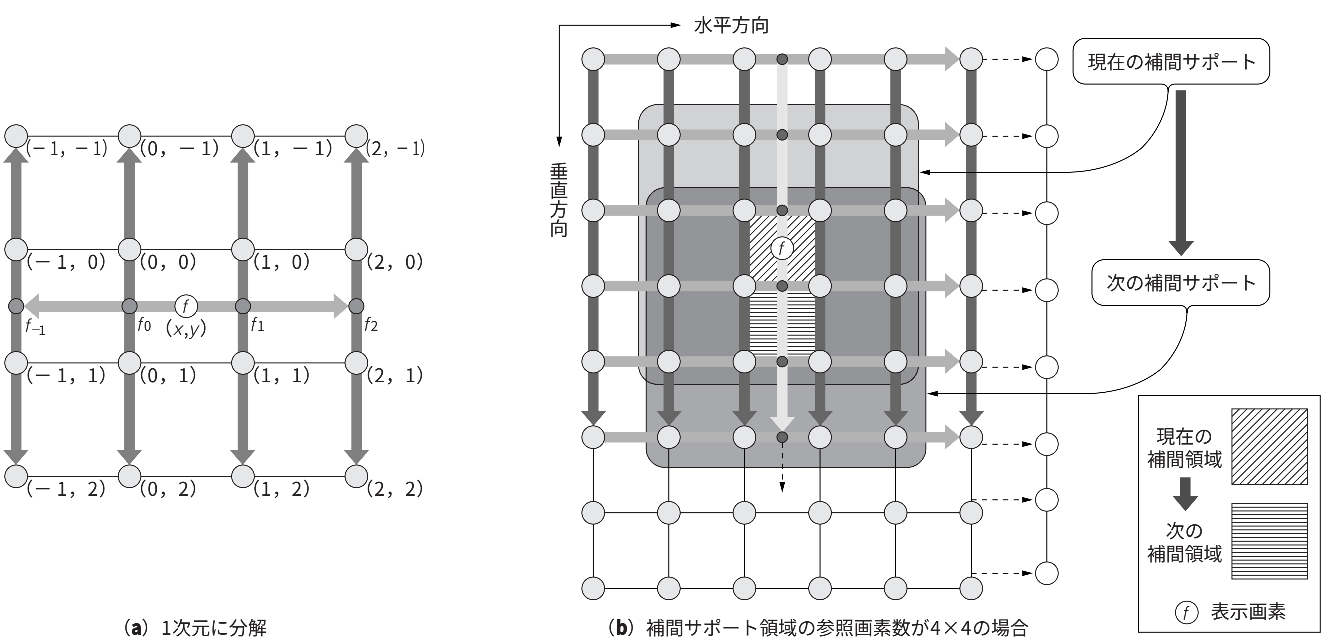


図2　参照画素点を有効に使いながら，表示点を補間演算する制御方式

（a）はバイキュービック補間のために垂直・水平方向に分解して3次補間を適用したもの．2次元のままで補間計算を扱うと複雑な式になる．垂直方向と水平方向に1次元分解することで，1次元のキュービック補間方式の単純な式の繰り返しで2次元化できる．（b）は，現在の補間サポート領域の参照画素点を有効に使うため，次の補間サポート領域に移る前に現在の補間領域に入るスケーリング表示画素点のすべてを求めるように制御する

●2次元補間は$n\times n$参照画素のサポート領域ごとに制御

以上で説明したのは，1次元（方向）の補間の場合です．画像は2次元のため，**図2**（**a**）に示すように，垂直方向と水平方向に分解して補間します（詳細は本連載第1回を参照）．

図2（**b**）に，補間サポート領域の参照画素が4×4の場合を示します．垂直，水平のどちらかの方向をまず補間します．例えば，水平方向を先に補間するとしましょう．このとき，以下のような流れで補間が行われます．

1）補間サポート領域にある水平4本の各参照画素4個に対して補間を行う

2）その結果を一時的に格納しておく

3）2）を垂直方向の参照画素4個に割り当てる

4）それらを用いて現在の表示画素*f*の補間を行う

表示画素fの座標位置からもっとも近い4個の原画素で囲まれる領域を「補間領域」と呼びます．参照する原画像の取り込みを効率良く行うには，現在の表示画素fの座標位置を含む補間領域内の表示画素点のすべてについて，ついでに演算してしまうことです．そして，次に表示するべき画素点が現在の補間領域からはみ出したら，新たに必要な補間サポート領域の参照画素を原画素から取り出し，不要になった補間サポート領域の参照画素を捨て，補間サポート領域を全体的にシフトして次の補間領域に移ります．

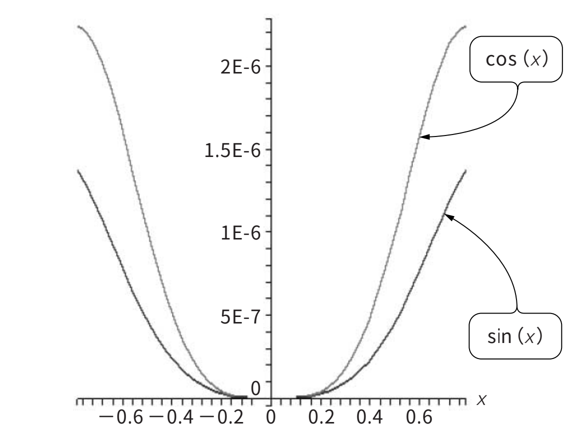
写真1は，前回解説したバイキュービック・スプライン（4×4画素参照）補間法を手っ取り早くデモンストレーシ

COLUMN ゾーンプレートと三角関数演算

画像スケーリング変換表示において，画質を評価するためにゾーンプレートがよく使われます．「エイリアス」といわれるひずみがある場合，ゾーンプレートを用いることで顕著に表示結果に現れるからです．表示結果には，ディジタルからアナログへ変換したときのエイリアスものるので，評価のときはどちらの症状が現れているか気をつけなければなりません．現に，本記事のために実際にディスプレイ表示した結果を誌面に印刷表示するまでの過程で，別の手段によってスケーリング変換がかかっています．その影響がところどころに出ているので，その点に留意して本記事の画像を見てください．

ゾーンプレートの式は，モノクロ256階調（8ビット）の場合，次式によって求められます．

$$d=127.5\cos\left\{(2\pi/r)\left((x-a)^2+(y-b)^2\right)\right\}+127.5 \quad\cdots\cdots\cdots\cdots\text{(A-1)}$$



図A-1　cos（*x*）とsin（*x*）の近似式の誤差曲線
表示範囲は−π/4≦*x*≦π/4であり，両端でもっとも誤差が大きい

ここで，（a，b）は同心円の中心座標，rは最小同心円の半径，πは円周率です．cos関数が必要なので，ここでは三角関数の最良近似式による演算器を構成し，かつ線形補間演算をベースとする方式の例として取り上げます．関数の最良近似式$g(x)$を求める方法として，i次のチェビシェフ（chebyshev）多項式$T_i(x)$を経由して求める方法がよく使われます．ここで，

$$T_0(x)=1,\quad T_1(x)=x,\quad\cdots,\quad T_{n+1}(x)=2xT_n(x)-T_{n-1}(x)\quad (n=1,\ 2,\ \cdots)\cdots\cdots\text{(A-2)}$$

です．すなわち，−π/4≦x≦π/4において，cos（x）とsin（x）はそれぞれを近似するcosx（x），sinx（x）という関数を定義して，以下の式で最良近似されます．

$$\begin{cases}\cos\mathrm{x}(x)=0.76520T_0(x)-0.22981T_2(x)\\ \qquad +0.0049533T_4(x)-0.000041877T_6(x)\\ \qquad =1.0-0.50000x^2+0.041636x^4-0.0013401x^6\\ \sin\mathrm{x}(x)=0.88010T_1(x)-0.039127T_3(x)\\ \qquad +0.00049952T_5(x)-0.0000030152T_7(x)\\ \qquad =x-0.16667x^3+0.0083300x^5-0.00019297x^7\end{cases}\cdots\text{(A-3)}$$

ここでは10進5けたの精度で求めています．いま，3次多項式演算に持ち込むため，線形補間演算ベースの式に変形すると，

$$\begin{cases}\cos\mathrm{x}(x)=\left(\left(0.54030x^2+0.54164\left(1-x^2\right)\right)x^2+0.5\left(1-x^2\right)\right)x^2\\ \qquad +\left(1-x^2\right)\\ \sin\mathrm{x}(x)=x\left(\left(\left(0.84147x^2+0.84166\left(1-x^2\right)\right)x^2\right.\right.\\ \qquad \left.\left.+0.83333\left(1-x^2\right)\right)x^2+\left(1-x^2\right)\right)\end{cases}\cdots\cdots\cdots\text{(A-4)}$$

が得られます．xの値域が−π/4≦x≦π/4であるため，cosx（x）とsinx（x）を使って［0，2π］の範囲の象限を表現する必要があります．**図A-1**にcos（x）とsin（x）の近似式の誤差曲線を示します．

ョンする目的で，FPGA評価ボード上で簡易的に実現したものです．ゾーンプレート原画像の16×16倍までの任意拡大率を指定して，その拡大画像をバイリニア補間法とともに比較表示できます．液晶パネルの下側に表示されている2×2倍～16×16倍の拡大率スケール図の該当位置をタッチすることにより，任意拡大率を読み取ることができます．また，読み込んだ拡大率(scale factor：16.0)をパネルの中央部に表示しています．この拡大率をもとにして1/拡大率を計算します．このとき除算器が必要になりますが，今回用いた評価ボードではCPUコアをFPGAに組み込んで利用できるため，CPUの除算命令をそのまま利用しています．また，拡大率を一定のきざみで変えながらアニメーション・モードで表示することもできます．

2. 評価——キュービック補間で注意すること

各種補間法の画質評価については，すでに連載第1回の中(本誌2006年7月号，p.80の**図2**と**図3**)で紹介しましたが，その図では縮小の場合はあまり違いがわかりませんでした．拡大の場合，最近傍選択，バイリニア，バイキュービック・コンボリューション，バイキュービック・スプライン(4×4画素参照)，バイキュービック・スプライン(6×6画素参照)の順で画質が良くなっていました．

●3×3倍以降の画質はアルゴリズムに大きく依存

ここでは，拡大率の変化に注目して，**図3**に2×2倍，3×3倍，5×5倍，7×7倍を並べて比較してみました．バ

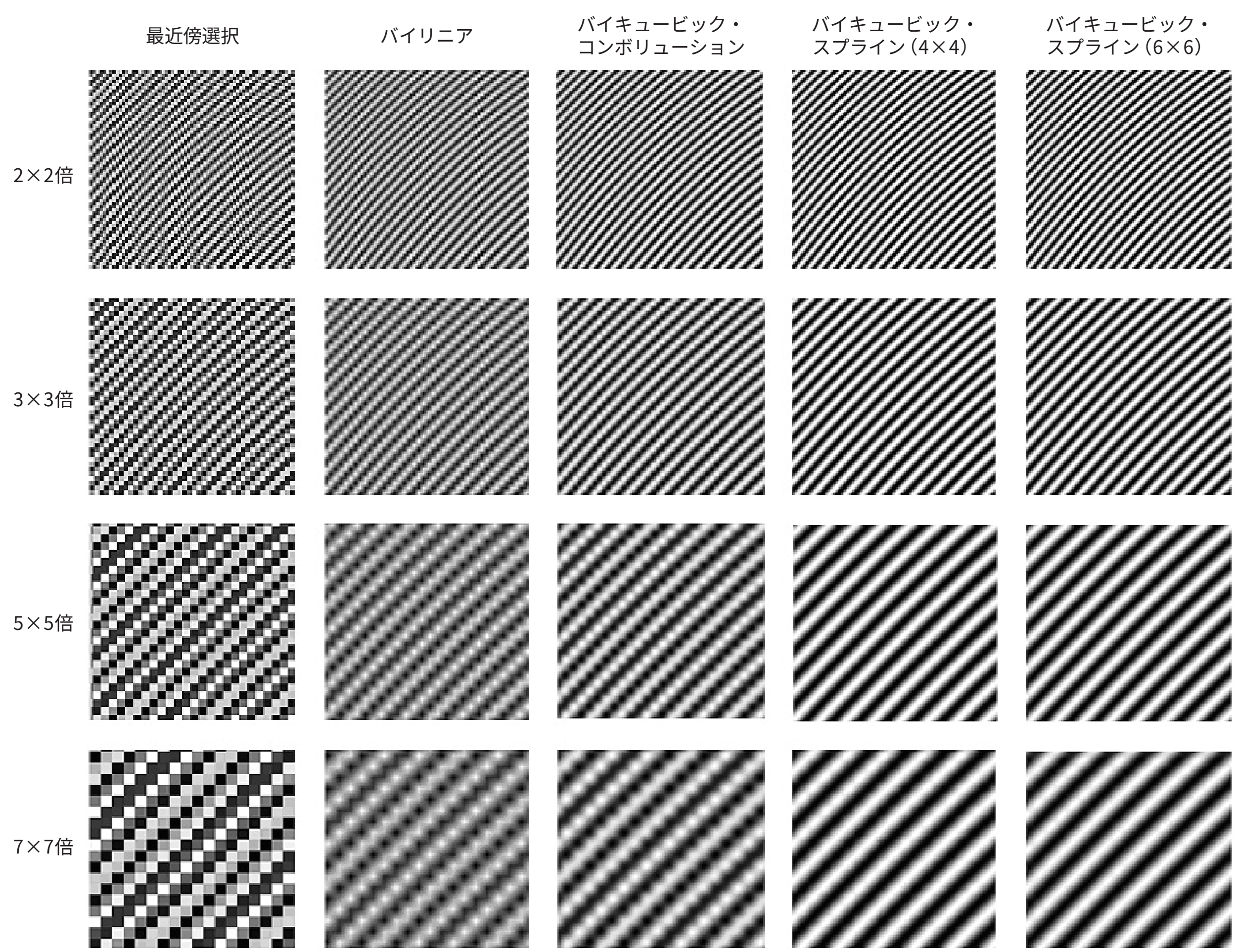


図3　各種補間法に基づく画像拡大スケーリング表示結果の比較

上から順に，各辺の拡大率が2，3，5，7倍(面積拡大率は4，9，25，49倍)の場合の表示比較．補間方式は左から，最近傍選択，バイリニア，バイキュービック・コンボリューション，バイキュービック・スプライン(4×4)，バイキュービック・スプライン(6×6)

イリニア法とバイキュービック法の違いは2×2倍あたりでは目立ちません．ディジタル・テレビにおいて，画素数の違う標準（1280×720画素）とフルハイビジョン（1920×1080画素）の画面を切り替えるときなどに使うのであれば，このあたりで十分です．

3×3倍程度からバイリニア法とバイキュービック法の違いが目立ってきます．ディジタル・カメラのディジタル・ズームで用いられているのがこのあたりまででしょうか．また，同じバイキュービック法でも，コンボリューション法とスプライン法の違いも出てきます．このため，医療用画像処理分野へ応用しようとしている技術者は，スプライン法に注目しています．5×5倍あたりからは，スプライン法の参照画素数による滑らかさの違いもわかるようになります．このように，従来のバイリニア補間では精度が不足すると考えられる応用分野では，バイキュービック補間の採用が考えられています．

ここで気をつけなければならないことがあります．それは，今回スケーリングの比較表示に用いたゾーンプレート画像は，人工的に生成描画したものだという点です（p.137のコラム「ゾーンプレートと三角関数演算」を参照）．

一般にディジタル・カメラ入力によって取得した画像の場合，**図4**に示す単板カラー撮像素子のBayer Patternと呼ばれている配列の知識が必要です．通常，欠損している位置の色データは，カメラ装置内で隣接画素間の平均から求めているか，あるいは独自の方法で求めているため，キュービック補間を行っても意味のある効果が得られるとはかぎりません．RAW（生）データ・モードをサポートしている最近のカメラであれば，Bayer Pattern配列のままの生データが取り出せるので，各R，G，B画素の正しい座標位置に基づいたキュービック補間を導入する必要があるでしょう．

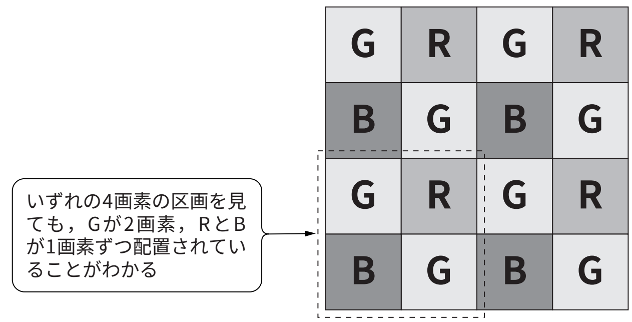


図4　単板カラー撮像素子のBayer Pattern配列
4画素の区画の中に，RGBを重ならないように配列している．G（緑）を2個，R（赤）とB（青）を1個ずつ配置している

とのむら・もとのぶ
大日本印刷（株）電子デバイス事業部 電子デバイス研究所

<著者プロフィール>
外村元伸．近くの公園にひまわりの苗が多数植えてあります．この記事をみなさんが読まれているころには，大きく咲き誇っていることでしょう．ひまわり畑の写真を撮って，拡大イメージのサンプルが作れたらと思っています．

COLUMN　前回の質問の解答(キュービック・スプライン補間係数の積和演算器の構成)

第2回の記事で，四つの画素値f_{-1}，f_0，f_1，f_2を参照するキュービック・スプライン補間法の係数a，b，c，dは，次の式で求められることを導きました．

$$\begin{cases} a = f_1 \\ b = \dfrac{1}{5}\left(3f_{-1} - 7f_0 + 12f_1 - 3f_2\right) \\ c = \dfrac{1}{5}\left(-4f_{-1} + 6f_0 + 4f_1 - f_2\right) \\ d = f_0 \end{cases} \qquad \cdots\cdots\text{(B-1)}$$

そして，$1/5 = \left[0.\dot{0}\dot{0}\dot{1}\dot{1}\dot{0}\dot{0}\dot{1}\dot{1}\right]\mathrm{b}$ は‘0’と‘1’が2個ずつ交互に繰り返す循環2進小数であり，1/5の整数倍であれば，例えば$3/5 = \left[0.\dot{1}\dot{0}\dot{0}\dot{1}\dot{1}\dot{0}\dot{0}\dot{1}\right]\mathrm{b}$などのように1/5のシフト演算で求められることを説明しました．また，具体的にどのように演算器を構成すればよいかをみなさんで考えてもらうことにしていました．

ここでは解答例として係数cについて，積和演算器の具体的な構成法を取り上げます．以下のように1/5の整数倍をA，画素値をYとします．

$$\begin{cases} A = \left[0.a_{-1}a_{-2}\,\mathrm{L}\;a_{-n}\right]\mathrm{b} = \displaystyle\sum_{i=1}^{n} a_{-i}2^{-i} \\ Y = \left[y_{n-1}y_{n-2}\,\mathrm{L}\;y_1y_0\right]\mathrm{b} = \displaystyle\sum_{j=0}^{n-1} y_j 2^j \end{cases} \qquad \cdots\cdots\text{(B-2)}$$

$$A \times Y = \sum_{i=1}^{n} a_{-i}2^{-i} \times \sum_{j=0}^{n-1} y_j 2^j = \sum_{i=1}^{n}\sum_{j=0}^{n-1} a_{-i}y_j 2^{j-i} \qquad \cdots\cdots\text{(B-3)}$$

そして，

$$-A = -1 + \sum_{i=1}^{n} \bar{a}_{-i}2^{-i} + 2^{-n}$$

であること(本誌2006年5月号，p.40の式(6)を参照)に注意すると，式(B-4)のようになります．

$$\begin{aligned} -A \times Y &= \left(-1 + \sum_{i=1}^{n} \bar{a}_{-i}2^{-i} + 2^{-n}\right) \times \sum_{j=0}^{n-1} y_j 2^j \\ &= -\sum_{j=0}^{n-1} y_j 2^j + \sum_{i=1}^{n}\sum_{j=0}^{n-1} \bar{a}_{-i} y_j 2^{j-i} + \sum_{j=0}^{n-1} y_j 2^{j-n} \qquad \cdots\cdots\text{(B-4)} \\ &= \left(-2^n + 1 + \sum_{j=0}^{n-1} \bar{y}_j 2^j\right) + \sum_{i=1}^{n}\sum_{j=0}^{n-1} \bar{a}_{-i} y_j 2^{j-i} + \sum_{j=0}^{n-1} y_j 2^{j-n} \end{aligned}$$

ここで，$\bar{a}_i = 1 - a_i$，$\bar{y}_j = 1 - y_j$です．さて，

$$c = \frac{4}{5}\left(-f_{-1} + f_1\right) - \frac{1}{5}f_2 + \left(1 + \frac{1}{5}\right)f_0 \qquad \cdots\cdots\text{(B-5)}$$

$$\begin{cases} \dfrac{1}{5} = \left[0.\dot{0}\dot{0}\dot{1}\dot{1}\right]\mathrm{b} = \displaystyle\sum_{i=1}^{\infty}\left(2^{1-4i} + 2^{-4i}\right) \\ -\dfrac{1}{5} = -\left[0.\dot{0}\dot{0}\dot{1}\dot{1}\right]\mathrm{b} = -\displaystyle\sum_{i=1}^{\infty}\left(2^{1-4i} + 2^{-4i}\right) \\ \qquad = -1 + \left[0.\dot{1}\dot{1}\dot{0}\dot{0}\right]\mathrm{b} = -1 + \displaystyle\sum_{i=1}^{\infty}\left(2^{3-4i} + 2^{2-4i}\right) \\ \dfrac{4}{5} = \left[0.\dot{1}\dot{1}\dot{0}\dot{0}\right]\mathrm{b} = \displaystyle\sum_{i=1}^{\infty}\left(2^{3-4i} + 2^{2-4i}\right) \\ -\dfrac{4}{5} = -\left[0.\dot{1}\dot{1}\dot{0}\dot{0}\right]\mathrm{b} = -\displaystyle\sum_{i=1}^{\infty}\left(2^{3-4i} + 2^{2-4i}\right) \\ \qquad = -1 + \left[0.\dot{0}\dot{0}\dot{1}\dot{1}\right]\mathrm{b} = -1 + \displaystyle\sum_{i=1}^{\infty}\left(2^{1-4i} + 2^{-4i}\right) \end{cases} \qquad \cdots\cdots\text{(B-6)}$$

となるので，式(B-5)の積和演算器を構成してみます．ここでは簡単化のために画素値は4ビットで，

$$f_i = \sum_{j=0}^{3} f_{ij}2^j$$

と表現します．式(B-3)～(B-5)を用いて，$n=4$の場合の具体的な式を求めると，**図B-1**に示す演算ブロックの記述が得られます．

$$\begin{aligned} &\left\{\left(-2^4 + 1 + \sum_{j=0}^{3} \bar{f}_{-1j}2^j\right) + \left(\sum_{j=0}^{3} f_{-1j}2^{j-3} + \sum_{j=0}^{3} f_{-1j}2^{j-4}\right) + \sum_{j=0}^{3} f_{-1j}2^{j-4}\right\} \\ &+\left\{\left(-2^4 + 1 + \sum_{j=0}^{3} \bar{f}_{2j}2^j\right) + \left(\sum_{j=0}^{3} f_{2j}2^{j-1} + \sum_{j=0}^{3} f_{2j}2^{j-2}\right) + \sum_{j=0}^{3} f_{2j}2^{j-4}\right\} \\ &+\left\{\sum_{j=0}^{n-1} f_{1j}2^{j-1} + \sum_{j=0}^{n-1} f_{1j}2^{j-2}\right\} + \left\{\sum_{j=0}^{n-1} f_{0j}2^j + \sum_{j=0}^{n-1} f_{0j}2^{j-3} + \sum_{j=0}^{n-1} f_{0j}2^{j-4}\right\} \end{aligned}$$

$$\sum_{j=0}^{3} f_{-1j}2^{j-2}$$

2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	
−1	$\bar{f}_{-13}$	$\bar{f}_{-12}$	$\bar{f}_{-11}$ f_{-13}	$\bar{f}_{-10}$ 1 f_{-12}	f_{-11}	f_{-10}			$-\frac{4}{5}f_{-1}$
−1	$\bar{f}_{23}$	$\bar{f}_{22}$ f_{23}	$\bar{f}_{21}$ f_{22} f_{23}	$\bar{f}_{20}$ 1 f_{21} f_{22}	f_{20} f_{21} f_{23}	f_{20} f_{22}	f_{21}	f_{20}	$-\frac{1}{5}f_2$
		f_{13}	f_{12} f_{13}	f_{11} f_{12}	f_{10} f_{11}	f_{10}			$\frac{4}{5}f_1$
	f_{03}	f_{02}	f_{01}	f_{00} f_{03}	f_{02} f_{03}	f_{01} f_{02}	f_{00} f_{01}	f_{00}	$\frac{6}{5}f_0$

図B-1　係数cの積和演算を簡単に実現する演算ブロック記述の構成例

画素値4ビットの例を示す．$1/5 = \left[0.\dot{0}\dot{0}\dot{1}\dot{1}\dot{0}\dot{0}\dot{1}\dot{1}\right]\mathrm{b}$ のように，非ゼロ項は半分であるため，専用演算器の規模は半分以下にできる